With SGI's MPT

Category: Porting to Pleiades

DRAFT

This article is being reviewed for completeness and technical accuracy.

Among the many MPI libraries installed on Pleiades, it is recommended that you start with SGI's MPT library.

The available SGI MPT modules are:

```
mpi/mpt.1.25
mpi-sgi/mpt.1.26
mpi-sgi/mpt.2.04.10789
```

There is no default MPT version set, but you are recommended to start with the MPT 2.04.10789 version by loading the *mpi-sgi/mpt.2.04.10789* module. You should load the same module when you build your application on the front-end node and also inside your PBS script for running on the back-end nodes.

Note: Pleiades uses an InfiniBand (IB) network for interprocess RDMA (remote direct memory access) communications and there are two InfiniBand fabrics, designated as *ib0* and *ib1*. In order to maximize performance, SGI advises that the *ib0* fabric be used for all MPI traffic. The *ib1* fabric is reserved for storage related traffic. The default configuration for MPI is to use only the *ib0* fabric.

Environment Variables

When you load an MPT module, several paths (such as CPATH, C_INCLUDE_PATH, LD_LIBRARY_PATH, etc) and MPT or ARRAYD related variables are set or modified. For example, with the mpi-sgi/mpt.2.04.10789 module, the following MPT and ARRAYD related variables are reset to some non-default values:

```
setenv        MPI_BUFS_PER_HOST 256
setenv        MPI_IB_TIMEOUT 20
setenv        MPI_IB_RAILS 2
setenv        MPI_DSM_DISTRIBUTE 0 (for Harpertown processors)
setenv        MPI_DSM_DISTRIBUTE 1 (for Nehalem-EP and Westmere-EP processors)
setenv        ARRAYD_CONNECTTO 15
setenv        ARRAYD_TIMEOUT 180
```

The meanings of these variables and their default values are:

- MPI_BUFS_PER_HOST

  Determines the number of shared message buffers (16 KB each) that MPI is to allocate for each host (i.e., Pleiades node used in the run). These buffers are used to send and receive long inter-host messages.

  Default: 96 pages (1 page = 16KB)
- MPI_IB_TIMEOUT

  When an IB card sends a packet it waits some amount of time for an ACK packet to be returned by the receiving IB card. If it does not receive one it sends the packet again. This variable controls that wait period. The time spent is equal to 4 * 2 ^ MPI_IB_TIMEOUT microseconds.

  Default: 18
- MPI_IB_RAILS

  If the MPI library uses the IB driver as the inter-host interconnect it will by default use a single IB fabric. If this is set to 2, the library will try to make use of multiple available separate IB fabrics (e.g. *ib0* and *ib1*) and split its traffic across them. If the fabrics do not have unique subnet IDs then the rail-config utility is required to have been run by the system administrator to enable the library to correctly use the separate fabrics.

  Default: 1

- MPI_DSM_DISTRIBUTE

  Activates NUMA job placement mode. This mode ensures that each MPI process gets a unique CPU and physical memory on the node with which that CPU is associated. This feature can also be overridden by using *dplace* or *omplace*. This feature is most useful if running on a dedicated system or running within a cpuset.

  Default: Enabled for MPT.1.26; Not Enabled for MPT.1.25

- ARRAYD_CONNECTTO

  Tuning this variable is useful when you want to run jobs through arrayd across a large cluster, and there is network congestion. Setting this variable to a higher value might slow down some array commands when a host is unavailable but it will help to prevent MPI start up problems due to connection time-out.

  Default: 5 seconds

- ARRAYD_TIMEOUT

Tuning this variable is useful when you want to run jobs through arrayd across a large cluster, and there is network congestion. Setting this variable to a higher value might slow down some array commands when a host is unavailable but it will help to prevent MPI start up problems due to connection time-out.

Default: 45 seconds

For more MPT related variables, read **man mpi** after loading an MPT module. Some of them may be useful for some applications or for debugging purposes on Pleides. Here are a few of them for you to consider:

- MPI_BUFS_PER_PROC

  Determines the number of private message buffers (16 KB each) that MPI is to allocate for each process (i.e. MPI rank). These buffers are used to send long messages and intrahost messages.

  Default: 32 pages (1 page = 16KB)

- MPI_IB_FAILOVER

  When the MPI library uses IB and a connection error is detected, the library will handle the error and restart the connection a number of times equal to the value of this variable. Once there are no more failover attempts left and a connection error occurs, the application will be aborted.

  Default: 4

- MPI_COREDUMP

  Controls which ranks of an MPI job can dump core on receipt of a core-dumping signal. Valid values are *NONE, FIRST, ALL*, or *INHIBIT. NONE* means that no rank should dump core. *FIRST* means that the first rank on each host to receive a core-dumping signal should dump core. *ALL* means that all ranks should dump core if they receive a core-dumping signal. *INHIBIT* disables MPI signal-handler registration for core- dumping signals.

  Default: FIRST

- MPI_STATS (toggle)

  Enables printing of MPI internal statistics. Each MPI process prints statistics about the amount of data sent with MPI calls during the MPI_Finalize process.

  Default: Not enabled

- MPI_DISPLAY_SETTING

  If set, MPT will display the default and current settings of the environmental variables controlling it.

  Default: Not enabled

- MPI_VERBOSE

  Setting this variable causes MPT to display information such as what interconnect devices are being used and environmental variables have been set by the user to non-default values. Setting this variable is equivalent to passing mpirun the -v option.

  Default: Not enabled

Building Applications

Building MPI applications with SGI's MPT library simply requires linking with -lmpi and/or -lmpi++. See the article SGI MPT for some examples.

Running Applications

```
MPI executables built with SGI's MPT are not allowed to run on the
Pleiades front-end nodes.
```

You can run your MPI job on the back-end nodes in an interactive PBS session or through a PBS batch job. After loading an MPT module, use **mpiexec**, not mpirun, to start your MPI processes. For example:

```
#PBS -lselect=2:ncpus=8:mpiprocs=4:model=har
....
module load mpi-sgi/mpt.2.04.10789
mpiexec -np N ./your_executable
```

The -np flag (with *N* MPI processes) can be omitted if the value of *N* is the same as the product of the value specified for *select* and that specified for *mpiprocs*.

Performance Issues

On Nehalem-EP and Westmere-EP nodes, if your MPI job uses all the processors in each node (i.e, 8 MPI processes/node for Nehalem-EP and 12 MPI processes/node for Westmere-EP), pinning MPI processes greatly helps the performance of the code. SGI's mpi-sgi/mpt.2.04.10789 will pin processes by default by setting the environment variable MPI_DSM_DISTRIBUTE to 1 (or true) when jobs are run on the Nehalem or Westmere nodes. On Harpertown nodes, setting MPI_DSM_DISTRIBUTE to 1 is not recommended due to a processor labeling issue.

If your MPI job do not use all the processors in each node, it is recommended that you disable MPI_DSM_DISTRIBUTE by

```
setenv MPI_DSM_DISTRIBUTE 0
```

and let the Linux kernel decide where to place your MPI processes. If you want to pin processes explicitly, you can use *dplace*. Beware that with SGI's MPT, only 1 shepherd process is created for the entire pool of MPI processes and the proper way of pinning using *dplace* is to skip the shepherd process. In addition, knowledge of the processor labeling in each processor type is essential when you use *dplace*. Below are the recommended ways of pinning an 8 MPI process job with every 4 processes on 4 processor cores of a node:

- Harpertown

```
mpiexec -np 8 dplace -s1 -c2,3,6,7 ./your_executable
```

- Nehalem-EP

```
mpiexec -np 8 dplace -s1 -c2,3,4,5 ./your_executable
```

- Westmere-EP

```
mpiexec -np 8 dplace -s1 -c4,5,6,7 ./your_executable
```

Further information about pinning can be found here.

Article ID: 100
Last updated: 03 Oct, 2011
Computing at NAS -> Porting & Developing Applications -> Porting to Pleiades -> With SGI's MPT
http://www.nas.nasa.gov/hecc/support/kb/entry/100/?ajax=1